1. (Cancelled)

2. (Currently amended) A computerized method for sequencing and reassembling messages from protocol data units exchanged in a communications channel between two computers, the ~~The~~ method ~~of claim 1~~ comprising:

creating a protocol flow object to represent each protocol layer used by the communications channel, each protocol flow object having a circuit element associated with each transmission direction in the channel;

arranging the protocol flow objects in a logical tree structure that mirrors a hierarchy for the protocol layers;

creating circuit flow objects for each protocol layer to represent the protocol data units for the protocol layer immediately higher in the hierarchy, wherein creating the circuit flow objects for each protocol layer comprises:

creating the circuit flow objects for the protocol flow object at the bottom of the tree structure by extracting data from the protocol data units for the protocol layer lowest in the hierarchy; and

creating the circuit flow objects for the remaining protocol flow objects in the tree structure by extracting data from the circuit flow objects linked to the protocol flow object immediately lower in the tree structure;

associating a transmission direction with each circuit flow object;

linking each circuit flow object for a protocol layer to the circuit element of the representative protocol flow object that matches the transmission direction associated with the circuit flow object;

sequencing the circuit flow objects linked to a particular protocol flow object when specified by the protocol layer represented by the particular protocol flow object; and

reassembling the messages from the circuit flow objects linked to the protocol flow object at the top of the tree structure.

3. (Currently amended) The method of claim ~~1~~ 2, wherein a circuit flow object comprises a vector list to represent fragmented data.

4. (Original) The method of claim 3, wherein a vector list comprises a vector specifying a protocol data unit number, a length value, and an offset value for each fragment of the fragmented data.

5. (Original) The method of claim 4 further comprising:

reassembling the fragmented data in accordance with the vectors in a vector list.

6. (Currently amended) The method of claim ~~1~~ 2 wherein the protocol flow objects are created in order from the bottom to the top of the hierarchy.

7. (Original) The method of claim 6, wherein the circuit flow objects for a current protocol flow object are created before creating the protocol flow object for the protocol layer immediately above the current protocol flow object in the hierarchy.

8. (Currently amended) The method of claim ~~1~~ 2 wherein arranging the protocol flow objects into a logical tree structure comprises:

creating multiple branches in the tree structure when a plurality of protocol layers are immediately above a current protocol layer in the hierarchy, each branch corresponding to one of the plurality of protocol layers.

9. (Currently amended) The method of claim ~~1~~ 2 further comprising:

determining the protocol layers in the hierarchy.

10. (Currently amended) The method of claim ~~1~~ 2 further comprising:

storing the protocol flow objects and the circuit flow objects in a flow object database.

11. (Cancelled)

12. (Currently amended) ~~The~~ A computer-readable medium ~~of claim 11~~ having ~~further~~ computer-executable instructions to a cause a computer to perform a method comprising:

creating a protocol flow object to represent each protocol layer used by a communications channel, each protocol flow object having a circuit element associated with a transmission direction in the channel;

arranging the protocol flow objects in a logical tree structure that mirrors a hierarchy for the protocol layers;

creating circuit flow objects for each protocol layer to represent the protocol data units for the protocol layer immediately higher in the hierarchy wherein creating the circuit flow objects for each protocol layer comprises:

creating the circuit flow object for the protocol flow object at the bottom of the tree structure by extracting data from the protocol data units for the protocol layer lowest in the hierarchy; and

creating the circuit flow objects for the remaining protocol layers by extracting data from the circuit flow objects linked to the protocol flow object immediately lower in the tree structure;

associating a transmission direction with each circuit flow object;

linking each circuit flow object for a protocol layer to the circuit element of the representative protocol flow object that matches the transmission direction associated with the circuit flow object;

sequencing the circuit flow objects linked to a particular protocol flow object when specified by the protocol layer represented by the particular protocol flow object; and

reassembling the messages from the circuit flow objects linked to the protocol flow object at the top of the tree structure.

13. (Currently amended) The computer-readable medium of claim ~~11~~ 12 having further computer-executable instructions comprising:

creating a circuit flow object as a vector list to represent fragmented data.

14. (Original) The computer-readable medium of claim 13 having further computer-executable instructions comprising:

creating a vector list from a plurality of vectors, each vector specifying a protocol data unit number, a length value, and an offset value for a fragment of the fragmented data.

15. (Original) The computer-readable medium of claim 14 having further computer-executable instructions comprising:

reassembling the fragmented data in accordance with the vectors in a vector list.

16. (Currently amended) The computer-readable medium of claim ~~11~~ 12 having further computer-executable instructions comprising:

creating multiple branches in the tree structure when a plurality of protocol layers are immediately above a current protocol layer in the hierarchy, each branch corresponding to one of the plurality of protocol layers.

17. (Currently amended) The computer-readable medium of claim ~~11~~ 13 having further computer-executable instructions comprising:

determining the protocol layers in the hierarchy.

18. (Currently amended) The computer-readable medium of claim ~~11~~ 12 having further computer-readable instructions comprising:

storing the protocol flow objects and the circuit flow objects in a flow object database.

19-23. (Cancelled)

24. (Currently amended) A computerized system comprising:

a processor;

a memory coupled to the processor through a bus;

a computer-readable medium coupled to the processor through the bus;

a plurality of protocol interpreters stored on the computer-readable medium for execution by the processor; and

a decode engine executed from the computer-readable medium to cause the processor to

create protocol flow objects representing protocol layers, each protocol flow object having a circuit element associated with each transmission direction in the channel, ~~and~~

create circuit flow objects representing data flows at the protocol layers, each circuit flow object for a protocol layer linked to the circuit element of the representative protocol flow object that matches the transmission direction associated with the circuit flow object, wherein creating the circuit flow objects comprises:

creating a circuit flow object for the protocol flow object at the bottom of a tree structure by extracting data from the protocol data units for the protocol layer lowest in a hierarchy of the protocol layers; and

creating circuit flow objects for the remaining protocol flow objects in the tree structure by extracting data from the circuit flow objects linked to the protocol flow object immediately lower in the tree structure,

extract data from the circuit flow objects representing protocol data units at a particular protocol layer as directed by one of the protocol interpreters,

sequence the circuit flow objects linked to a particular protocol flow object representing the protocol data units at a particular protocol layer if directed by one of the protocol interpreters, and

reassemble messages from the circuit flow objects linked to the protocol flow object at the tope of the tree structure representing the protocol data units at a particular protocol layer if directed by one of the protocol interpreters.

25. (Original) The computer system of claim 24, wherein the decode engine further causes the processor to store the protocol flow objects and circuit flow objects in a flow database, logically link the protocol flow objects into a hierarchical tree structure, and to logically link the circuit flow objects to the protocol flow objects.

26. (Original) The computer system of claim 24, wherein the decode engine further causes the processor to create a circuit flow object as a vector list to represent fragmented data.

27. (Original) The computer system of claim 26, wherein the decode engine further causes the processor to create a vector list from a plurality of vectors, each vector specifying a protocol data unit number, a length value, and an offset value for a fragment of the fragmented data.

28. (Original) The computer system of claim 27, wherein the decode engine further causes the processor to reassemble the fragmented data in accordance with the vectors in a vector list.

29-30. (Cancelled)

31. (New) A computerized method for sequencing and reassembling messages from protocol data units exchanged in a communications channel between two computers, the method comprising:

creating a protocol flow object to represent each protocol layer used by the communications channel, wherein the protocol flow objects are created in order from a bottom to a top of hierarchy determined by the protocol layers, and each protocol flow object has a circuit element associated with each transmission direction in the channel and has a data structure including a key field containing data representing an identifier for a connection between two computers at one of the protocol layers, the circuit elements including a primary circuit element containing data representing a link to a series of protocol data units flowing in one direction in the connection identified by the key field, and an alternate circuit element containing data representing a link to a series of protocol data units flowing in an opposite direction in the connection identified by the key field, wherein the links comprise hash tables for identifying the series of data units;

arranging the protocol flow objects in a logical tree structure that mirrors a hierarchy for the protocol layers, the logical tree structure having multiple branches when

a plurality of protocol layers are immediately above a current protocol layer in the hierarchy, each branch corresponding to only one of the plurality of protocol layers, wherein the logical tree structure comprises a plurality of entries, each entry comprising a protocol field containing data representing the identifier for one of the plurality of protocol flow objects, the protocol fields including a lower protocol field containing data representing the identifier for the protocol flow object immediately lower in the hierarchy relative to the protocol flow object identified by the protocol field, and a higher protocol field containing data representing the identifier for the protocol flow object immediately higher in the hierarchy relative to the protocol flow object identified by the protocol field;

creating circuit flow objects for each protocol layer to represent the protocol data units for the protocol layer immediately higher in the hierarchy, at least one of the circuit flow objects comprising a vector list to represent fragmented data to indicate how to reassemble the fragmented data, the vector list comprising a vector specifying a protocol data unit number, a length value, and an offset value for each fragment of the fragmented data, and wherein creating the circuit flow objects for each protocol layer comprises:

creating the circuit flow objects for the protocol flow object at the bottom of the tree structure by extracting data from the protocol data units for the protocol layer lowest in the hierarchy; and

creating the circuit flow objects for the remaining protocol flow objects in the tree structure by extracting data from the circuit flow objects linked to the protocol flow object immediately lower in the tree structure, with the circuit flow objects for a current protocol flow object being created before creating the protocol flow object for the protocol layer immediately above the current protocol flow object in the hierarchy;

associating a transmission direction with each circuit flow object;

linking each circuit flow object for a protocol layer to the circuit element of the representative protocol flow object that matches the transmission direction associated with the circuit flow object;

sequencing the circuit flow objects linked to a particular protocol flow object when specified by the protocol layer represented by the particular protocol flow object; and

reassembling the messages from the circuit flow objects linked to the protocol flow object at the top of the tree structure,

wherein the protocol flow objects and the circuit flow objects are stored in a flow object database and the method is invoked through an application program interface specifying at least five arguments selected from the group consisting of an hInterp argument, a uOffset argument, a uTotalLength argument, a uFragLength argument, a ulSequence argument, a ulID argument, a uPostFlags argument, and a uProtoID argument.